Resumo Alargado

Rafael Ribeiro, Daniel Silvestre, and Carlos Silvestre

Abstract— **This paper addresses the problem of having a multi-agent system converge to multiple dynamic rendezvous areas while creating and maintaining dynamic formations, in generally disconnected network topologies. There is no assumption on the connectedness of the network topology, which is unknown to all agents. The main proposed algorithm is fully-decentralized and considers non-communicating agents with localization and measuring capabilities. We also present a partially-decentralized solution for communicating agents with no localization capabilities, which do not have an objective to create and maintain formations. The implementations consist of improved flocking-based movement algorithms tailored to the proposed scenario coupled with a utility function defining the mission plane and mechanisms to prevent agent-agent and agent-obstacle collisions. The performance of the algorithm is presented through simulations for a variety of environments. These empirical results show the agents rendezvous to the multiple dynamic rendezvous areas in the presence of undesirable areas, static environmental obstacles, and arbitrary changes in the utility function, with varying degrees of efficiency.**

I. Introduction

The fully-decentralized solution proposed in this paper is currently under review in [1]. As the associated dissertation presents an additional algorithm - a partially-decentralized solution (published in [2] and under review in [3]) - Section III was added to this paper regarding its implementation and system assumptions, from which the fully-decentralized solution was created.

The rendezvous problem, defined as a group of agents attempting to converge to the same area in the mission plane, is a central issue when establishing a swarm-like network of agents to perform a given assignment. This paper addresses the rendezvous problem for multiple dynamic rendezvous areas for agents in formation by proposing an algorithm that accounts for sensor noise, no communication capabilities, and an unknown network topology. The system poses no assumptions on the connectivity of the initial configuration, which will be disconnected in the general case, meaning that the underlying graph is composed of multiple clusters. The rendezvous areas are classified as dynamic because their values and positions are time-varying, which can account for scenarios such as exploration of a new location or surveillance.

The envisioned scenario corresponds to a mission to be carried out by identical, low cost, autonomous agents which are equipped with localization and measuring sensors to be aware of their position and velocity, and those of their neighbors. Because the positioning data is measured by the agents in a relative frame of reference, the proposed system is ideal for operations in GPS-denied environments, such as open-space indoor navigation or search and rescue missions in bare locations, and avoids the requirement of a standardized coordinate system. Instead of a consensus-based approach, the behavior of the agents uses flocking-based movement rules, with added components tailored to our scenario. We also introduce a mechanism, with theoretical guarantees, to avoid collisions between agents.

In the literature, the usual approach for a group of mobile agents to achieve rendezvous is based on consensus: agents obtain the position of their neighbors, calculate the average position, and move towards it. Consequently, this method only considers one rendezvous area - the average position of the cluster - which has no additional value than being the closest position. Moreover, this approach requires the network topology to be initially connected. There are numerous examples of consensus algorithms being proposed in the literature to address different scenarios: switching topology and time-delays [4]; networks with stochastic asymmetric communications [5], [6]; networks with communication link failures [7], [8]; networks with quantized data transmission [9]; event-triggered [10] and self-triggered [11] control.

The problem of rendezvousing to multiple areas is addressed in [12], which proposes a leader/follower approach. Agents are divided into groups (one group per rendezvous point) and establish a hierarchical tree structure in which the leader of each group is aware of the target position and the follower agents are only aware of the position of their immediate leaders.

In order to rendezvous in a desired, unknown position, [13] assumes a scalar field defined by a function with a single maximum. Each agent is equipped with sensors to sense the value of the scalar field at its position, and communication equipment that transmits the values to its neighbors. With the received values, the agents calculate the desired movement direction based on the relative difference of these values. As a result, agents move towards better placed neighbors. Similarly to consensus approaches, [13] only considers one desired rendezvous area. A secondary objective for our agents is to converge to the most desirable rendezvous area. Desirability is defined by a utility function that attributes a numerical value to each position of our mission plane regarding its

R. Ribeiro is with the Instituto Superior Técnico, Universidade de Lisboa, Lisbon, Portugal, rribeiro@isr.ist.utl.pt

D. Silvestre is with the Department of Electrical and Computer Engineering of the Faculty of Science and Technology of the University of Macau, Macau, China, and with the Institute for Systems and Robotics, Instituto Superior Técnico, Universidade de Lisboa, Lisbon, Portugal., dsilvestre@isr.ist.utl.pt

C. Silvestre is with the Department of Electrical and Computer Engineering of the Faculty of Science and Technology of the University of Macau, Macau, China, on leave from Instituto Superior Técnico, Universidade de Lisboa, Lisboa, Portugal, csilvestre@umac.mo

This work was partially supported by the project MYRG2018-00198-FST of the University of Macau; by the Portuguese Fundação para a Ciência e a Tecnologia (FCT) through Institute for Systems and Robotics (ISR), under Laboratory for Robotics and Engineering Systems (LARSyS) project UIDB/50009/2020.

quality for the overall objective. The rendezvous targets are the maxima of the utility function. Unlike [13], we do not make the assumption of a single stationary point. Moreover, there does not exist communication between the nodes where each agent broadcasts its value. In contrast, in our method, each agents is equipped with sensors to measure the utility value in their positions and those of their neighbors. In [13], the algorithm has an attraction force towards agents in positions with a higher field value. We adopt a similar solution by adding two movement components on top of the original flocking rules:

1) an attraction force towards agents in higher-utility positions and a repulsion force against agents in lower-utility positions;
2) an attraction force in the direction that maximizes the gradient of the utility function evaluated at the agent position. This component is only used if the function is differentiable at the required point.

In the literature, the solutions for the problem of formation construction and maintenance can be divided into three general categories: i) physical leader, ii) virtual leader, and iii) distance-based formations. As an example of i), [14] addresses the problem of tracking a target trajectory while maintaining the desired formation. The target position is assumed to be known to all agents. The formation slots are defined relative to the target, and are predetermined and pre-assigned, meaning that the structure does no adapt to agents entering or leaving. The work in [15] uses a structure defined by a hierarchical leadership. Each agent follows its immediate leader (according to the defined hierarchy) by maintaining a predetermined relative position. The assignment is executed through consensus based on the agents state. This approach has limited scalability as it would require a new consensus procedure whenever an agent joins or leaves the formation. A case of ii) can be found in [16], where a virtual trajectory is assumed to be available as part of the mission data. Such a procedure conflicts with our scenario in the sense that the trajectory will be determined by sensing the utility function. In iii), the formation structure is defined by a distance matrix, such as in [17]. Agents move in the direction that minimizes the error between the entries in the distance matrix and the actual distances to their neighbors. In [18], the formation has a desired velocity, known by the formation agents: this implementation can be considered a leader/follower approach. Generally, the literature examples consider predetermined formation structures with pre-assigned slots, which results in no new agents being allowed to participate in the formation and the structure not adapting to agents leaving.

Another essential feature to be considered in rendezvous algorithms is the need to take into account noisy measurements while having guarantees that there will be no collisions. In [19], this consensus problem for a group of agents with uncertain positions is tackled: the positions are measured as arbitrary polytopes, and it is shown the agents can make predictions of worst-case positions. In this paper, similarly to [19], the estimate of an agent state is assumed to be given as a convex polytope containing all the possible valid positions given the measurements. Using Set-valued Observers (SVOs) makes possible the propagation of the estimate for the location of each agent using its dynamical model. A significant direct result is that the collision avoidance technique, included in the proposed algorithm, is supported by theoretical guarantees.

The algorithm proposed in this paper can be seen as a generalization from the flocking rules proposed [2], [20] by having a completely decentralized control law and forcing the agents to establish loose formation whenever not near maximum utility positions. Moreover, by equipping sensors within the mobile agents, the work presented herein is capable of considering mission setups where the agents form a true swarm.

The remainder of this paper is organized as follows. Section II presents background material and defines the tackled problem. The partially-decentralized rendezvous algorithm is given in Section III and the fully-decentralized solution in Section IV. Section V presents results regarding the convergence and guarantees of the fully-decentralized method, while simulations are provided in Section VI. Conclusions and directions for future work are offered in Section VII.

II. Problem statement

The agents traverse a mission plane which is responsible for representing the rendezvous (desired) areas, undesired areas, and unauthorized areas (obstacles) through its utility function, which attributes a numerical value to each position of the plane regarding its quality for the mission objective. In order to simplify the complexity associated with the algorithm, we have assumed that the function is given as the sum of paraboloids, each representing an area. This allows for the utility values of the areas to be dynamic and change due to the agent exploration: whenever an agent is not exploring a rendezvous area, its utility value increases (until it reaches a maximum); whenever an agent is exploring a rendezvous area, the value decreases until it reaches a minimum threshold - at which point it is removed from the mission plane. In this paper, we assume the utility function h is time-varying. Function h should take into account the boundaries and desired and undesirable areas, as well as unauthorized zones. Each area's definition is given by:

$$\frac{\kappa}{(\phi x)^2 + (\tau y)^2 + \omega} \tag{1}$$

where $\kappa, \omega \in \mathbb{R}^+_{\neq 0}$ modulate the paraboloid's height, $\phi, \tau \in \mathbb{R}$ modulate its slope. However, we remark to the reader that other types of functions could be used or even data based from samples. In doing so, a change in one of the paraboloids can be done to remove or add areas to the utility function in a fairly straightforward way.

The n agents composing the system, represented by the node set $\mathcal{V}$, are equipped with sensors to measure the value for any position within a sensing radius. Moreover, nodes have access to the position and velocity of all other agents within the sensing radius. Therefore, the neighbor relation definition is based on the sensing radius. Agent i's neighbors are all agents within SR distance units. Formally, the neighbor set of agent i at time step k is defined as:

$$\mathcal{N}_i(k) = \{j \in \mathcal{V} : \parallel q_i(k) - q_j(k) \parallel < SR\} \tag{2}$$

with $q_i(k)$ and $q_j(k)$ being the exact positions of agents i and j at k.

Given that sensors are corrupted by noise, a node i gets a set-valued estimate that is assumed to be given as a convex polytope $\mathcal{X}_i(k)$ provided by a set-membership filter such as Set-Valued Observers (SVOs) in [21]. We remark that, in general, the centroid of the set does not correspond to the true value.

The dynamics for each agent is assumed to be given by a double integrator in discrete-time:

$$\begin{aligned} q_i(k+1) &= q_i(k) + v_i(k), \\ v_i(k+1) &= v_i(k) + u_i(k), \\ i &= 1, 2, \ldots, n \end{aligned} \tag{3}$$

where q_i, v_i and u_i represent agent i's exact position, velocity, and control input, respectively.

III. Partially-decentralized Control Solution

In the partially-decentralized solution, the agents do not have localization or measuring sensors, and rely on broadcasts from control/communication towers to become aware of their position, velocity, and utility value, and those of their neighbors.

The control towers are responsible for transmitting the agent states and compose a message comprising of the utility value, position and velocity estimates of each agent in a communication cone. Since transmission is directional, each tower will send a message to a given node in a round robin fashion, which is received in a circular segment of fixed radius originating at the tower's position. In doing so, power is saved in the communication and results in a better use of the shared medium. Notice that for large networks the sent messages in each direction are reduced in size since a full broadcast is avoided. In a swarm formation of autonomous robots, the towers can also be implemented as more sophisticated and expensive mobile agents carrying extra sensors and communication capabilities. For instance, towers can be equipped with Radar or Lidar systems in order to take measurements of the agents.

Due to the absence of localization sensors, the agents rely on the messages from the towers to determine their location and calculate the proposed control law. These measurements are imperfect, corrupted by noise, meaning that instead of accessing the true state, the towers determine a convex polytope - $\mathcal{X}_i(k)$ - containing all possible positions for agent i at time k, whose centroid $c_i(k)$ will typically not correspond to the true position.

In general, the network topology will be disconnected and composed of a finite number of clusters at any given time k. We do not assume knowledge of the network topology nor can agents determine n. Agent i will only have access to its neighbor set $\mathcal{N}_i(k)$ by listening to the communication channel. Any two agents i and j are considered to be neighbors if they are bounded by the same communication cone.

Apart from the existence of communication towers (which replace the sensing radius), the assumptions from Section II hold.

The movement algorithm implemented for the partially-decentralized solution was the basis for the solution presented in Section IV. Except for the *Formation* component and associated formation assembly (described in Section IV-B), the control law used by the agents is the same, including the collision avoidance methods.

IV. Decentralized Control Solution

In this paper, in order to design a decentralized algorithm, it was followed the idea of defining the control actuation based solely on local information that can be accessed if the nodes have the considered sensors. The proposal extends the original flocking rules with three additional components tailored for the proposed environment. Later, collision avoidance will be addressed as an overlay to the current actuation. We first review the three traditional rules followed by the proposed ones.

Separation Component The *Separation* component is responsible for repelling agents as a naive implementation of a collision avoidance mechanism. Formally, $u_i^s(k)$ is defined as:

$$u_i^s(k) = \frac{1}{|\mathcal{N}_{i,d}(k)|} \sum_{j \in \mathcal{N}_{i,d}(k)} c_i(k) - c_j(k) \tag{4}$$

where $\mathcal{N}_{i,d}$ is the set of agent i's neighbors within d unit distances.

Cohesion Component The *Cohesion* component is responsible for decreasing the overall distance between neighbors and preventing agents to excessively separate. Formally, $u_i^c(k)$ is defined as:

$$u_i^c(k) = \frac{1}{|\mathcal{N}_i(k)|} \sum_{j \in \mathcal{N}_i(k)} c_j(k) - c_i(k) \tag{5}$$

In *Separation*, only neighbors closer than d are considered. This is justified by separating from distant agents being counterproductive to the mission objective of rendezvousing. As *Cohesion* aligns with this mission objective, all neighbors are considered.

Alignment Component The *Alignment* component causes a cluster of agents to align their direction vectors. Formally, $u_i^a(k)$ is defined as:

$$u_i^a(k) = \frac{1}{|\mathcal{N}_i(k)|} \sum_{j \in \mathcal{N}_i(k)} v_j(k) - v_i(k) \tag{6}$$

Attraction Component *Attraction*, along with *Utility* and *Randomness*, is a custom component, supplementary to the original flocking rules. It is responsible for moving clusters to higher-utility positions. For this purpose, each agent is attracted to neighbor higher-utility agents and repelled from neighbor lower-utility agents. A higher-utility agent is an agent

in a higher-utility position relative to the agent computing the control law. Formally, $u_i^{attr}(k)$ is defined as:

$$u_i^{attr}(k) = \frac{1}{|\mathcal{N}_i(k)|} \sum_{j \in \mathcal{N}_i(k)} (h(c_j(k)) - h(c_i(k)))(c_j(k) - c_i(k)) \tag{7}$$

where $h(c_i(k))$ and $h(c_j(k))$ are the utility function's values at agent i's and agent j's centroid positions, respectively.

Utility Component The *Utility* component is responsible for moving the agents in the direction that locally maximizes the utility function by following the gradient at the current position, which results in agents converging in higher-utility areas. When the utility function is not differentiable at the agent's position, this component has a zero-vector value. Formally, $u_i^u(k)$ is defined as:

$$u_i^u(k) = \nabla h(c_i(k)) \tag{8}$$

Formation Component The *Formation* component is responsible for moving the agents to their formation slots. Formally, $u_i^f(k)$ is defined as:

$$u_i^f(k) = f_i(k) - c_i(k) \tag{9}$$

where $f_i(k)$ is the position of agent i's slot in the formation, defined in Section IV-B.

Randomness Component In order to avoid indefinite movement deadlocks, the *Randomness* component, $u_i^r(k)$, adds a unit vector with a random angle to the agent acceleration vector.

A. Desired Movement Law

The desired movement law is constructed as a weighted average of the multiple components. Each weight translates to the importance of the respective component for the mission objective. For instance, if the mission objective is to explore the maximum number of rendezvous points without the need for agent convergence, then the *Utility* component's weight will be higher than its *Cohesion* counterpart.

Formally, the desired movement law, $\hat{u}_i(k)$, is defined with:

$$\begin{aligned}\hat{u}_i(k) = \theta \cdot u_i^s(k) + \beta \cdot u_i^c(k) + \gamma \cdot u_i^a(k) + \delta \cdot u_i^{attr}(k) \\ + \epsilon \cdot u_i^u(k) + \zeta \cdot u_i^f(k) + \eta \cdot u_i^r(k)\end{aligned} \tag{10}$$

where all weights $\theta, \beta, \gamma, \delta, \epsilon, \zeta, \eta \in \mathbb{R}^+$, with all component vectors normalized.

B. Formation Assembly

Each agent creates a virtual structure representing the desired formation - regular m-sided polygon - based on the number m of agents it assumes are participating in the physical formation, corresponding to its neighbors. Neighbor agents can have different neighbor sets, which can cause the virtual formation structure of two neighbor agents to be different. This problem is mitigated by the *Cohesion* component which decreases the probability of a cluster of agents having different neighbor sets, by decreasing the overall distance between agents in the cluster.

The formation slots correspond to the positions of the vertices of the m-sided polygon structuring the formation. The slots are assigned by each agent independently, by attributing each slot to the agent closest to it. This heuristic was chosen due to its simplicity and non-conflicting properties - it does not cause double-booking of slots.

C. Agent Collision Avoidance

The implemented collision avoidance mechanism has two stages: collision detection (Polytope Propagation and Planned Collision Detection) and movement restriction. The former calculates if the moving agent will collide with any of its neighbors if it moves the maximum allowed distance per update, D_{max}, in the desired direction. The latter clips the desired movement to prevent collisions.

Before detecting future collisions, the position-estimate polytopes of the neighbor agents have to be propagated using the dynamics such that they account for all positions to which the nodes could have moved up until the current time instant. This will result in enlarging the polytope to include any points within D_{max} distance units, for each time step, from the original polytope. This is essentially a Minkowski sum of the polytope with the interval $[-D_{max}, D_{max}]$ for each discrete time step.

The collision detection implementation utilizes the propagated polytopes, and detects possible collisions between every agent pair (i, j), with i being the calculating agent and $j \in \mathcal{N}_i(k) \setminus i$. A ray is cast from each vertex of agent i's polytope, with a length equal to the maximum allowed distance per update, and direction equal to the desired movement. An intersection between this ray and agent j's polytope represents a possible future collision. This ray casting process is repeated from the vertices of agent j's polytope, with rays in the opposite direction.

The final step of this phase is to calculate the minimum distance from the origin of each ray to its intersection with the opposite polytope, which represents the maximum distance, D, the agent can move in its desired direction without colliding with any agent j (from the (i, j) pairs used in the ray-tracing pair-wise detection process).

The implemented ray-tracing method for collision detection provides exact results in two-dimensional space. The first collision point between two convex polytopes corresponds to a vertex (from either polytope), and as such, by tracing the future path of each vertex and verifying if it intersects with another convex polytope, this method determines if there will be a collision.

In order to guarantee a movement with no agent-agent collisions, agent i's control input magnitude is culled based on D - the distance calculated in the previous step. A pair of neighbor agents sufficiently distanced to cause a possible collision will generally desire to move in a similar direction, mostly due to the *Utility* component, or in a direction that

separates them, mostly due to *Separation* and *Formation*. As such, agent i's movement distance is restricted, because, generally, the obstacle-agent in its path will eventually move.

D. Environmental Collision Avoidance

The proposed implementation considers two types of environmental obstacles: the mission plane perimeter walls and generalized static obstacles. The first type is avoided using a force field method, which consists of adding a virtual repulsion force to the perimeter walls, similar to the *Separation* component.

The method for avoiding the second type - presented in the remainder of this section - is analogous to the agent-agent collision avoidance method. Instead of maximizing the distance that results in a collision-free movement, this method prevents the agent from moving if it would result in a collision with an environmental obstacle, as it is not advantageous to the mission objective for an agent to minimize its distance to environmental obstacles. The environmental obstacles are static, so propagating the polytope is not required. To detect a collision with the environment, the agent calculates if the future position resulting from its desired movement is inside an unauthorized area. As such, when an agent detects its desired movement will result in a collision with an environmental obstacle, it stops. This movement restriction is justified by the two following general cases:

- if the agent does not have neighbors, its *Randomness* component will eventually cause it to move away from the obstacle.
- if the agent has neighbors, they will eventually attract it away from the obstacle.

Formally, the result of this method only has two possible values: $D_{env} \in [0, 1]$ if the movement will not result in a collision or $D_{env} = 0$, otherwise.

E. Final Control Law

The final control law will consider the agent's desired movement and the maximum distances allowed by the collision avoidance methods. The control input given to the agent will have the desired direction and a magnitude equal to the minimum between the resulting distances from the agent and environmental collision avoidance methods. Formally, $u_i(k)$ is defined as:

$$u_i(k) = \frac{\hat{u}_i(k)}{\|\hat{u}_i(k)\|} \cdot \min(D, D_{env}) \tag{11}$$

with $\hat{u}_i(k)$, D, and D_{env} being the desired control law, the maximum distance allowed by the agent-agent collision avoidance method, and the maximum distance allowed by the agent-obstacle collision avoidance method, respectively.

V. Convergence Analysis

In the previous section, we have constructed the actuation law as a sequence of terms that should guide each of the agents towards one of the objectives. These rules were inspired by the traditional mechanisms that are used to deterministically simulate flocks. In this section, we aim to provide a convergence analysis by viewing the overall system as a consensus algorithm driven by a signal that corresponds to a noisy gradient and a noisy desired velocity term to achieve the formation.

Theorem 1: Consider a network of n agents running the algorithm given in 11 for sufficiently small $\theta, \beta, \delta, \eta \in \mathbb{R}^+$ and positive constants γ, ϵ, ζ. Then, for each agent i, it will satisfies one of the following:

i) $\lim_{k\to\infty} \|c_i(k) - f_i(k)\| \leq \varphi_1(\mathrm{p}, \xi)$, where $\varphi_1(\mathrm{p}, \xi)$ is some constant dependent on the whole vector of parameters p and the noise ξ associated with the set-valued estimates;

ii) $\lim_{k\to\infty} \|x_i(k) - x^\star\| \leq \varphi_2(\mathrm{p}, \xi)$, where $\varphi_2(\mathrm{p}, \xi)$ is some constant dependent on the whole vector of parameters p and the noise ξ associated with the set-valued estimates, and $x^\star = \arg\max_{x\in\mathcal{C}} h(x)$ for some closed set $\mathcal{C}$ such that $x^\star$ does not belong to the boundary of $\mathcal{C}$.

Proof: Given that the statement of the theorem fixes a value for n, we can address the question with a single cluster of connected nodes and the same asymptotical results will hold when new nodes of a different cluster come in contact with the current one. Without the collision avoidance mechanism, the velocity vector of each node in the cluster is subject to the following dynamics:

$$\begin{aligned} v_i(k+1) &= v_i(k) + \gamma \frac{1}{|\mathcal{N}_i(k)|} \sum_{j\in\mathcal{N}_i(k)} v_j(k) - v_i(k) \\ &\quad + \epsilon\nabla h(c_i(k)) + \zeta\left(f_i(k) - c_i(k)\right) + \sigma_i^{\mathrm{p}}(k) \\ &= (1-\gamma)v_i(k) + \gamma \frac{1}{|\mathcal{N}_i(k)|} \sum_{j\in\mathcal{N}_i(k)} v_j(k) \\ &\quad + \epsilon\nabla h(c_i(k)) + \zeta\left(f_i(k) - c_i(k)\right) + \sigma_i^{\mathrm{p}}(k) \end{aligned} \tag{12}$$

where $\sigma_i^{\mathrm{p}}(k)$ can be seen as a perturbation associated with the remaining components and such that $\|\sigma_i^{\mathrm{p}}(k)\|$ is sufficiently small and dependent on the vector of parameters p that contains θ, β, $\cdots$. We can further define $v_i^f(c_i(k)) := f_i(k) - c_i(k)$ since this is essentially a velocity vector driving the agent from $c_i(k)$ towards its intended position in the formation $f_i(k)$. Thus, we can further simplify the expression as:

$$\begin{aligned} v_i(k+1) &= (1-\gamma)v_i(k) + \gamma \frac{1}{|\mathcal{N}_i(k)|} \sum_{j\in\mathcal{N}_i(k)} v_j(k) \\ &\quad + \epsilon\nabla h(c_i(k)) + \zeta v_i^f(c_i(k)) + \sigma_i^{\mathrm{p}}(k) \end{aligned} \tag{13}$$

Writing the above expression in vector form returns:

$$v(k+1) = W_\gamma v(k) + \epsilon\nabla\mathbf{h}(c(k)) + \zeta v^f(c(k)) + \sigma^{\mathrm{p}}(k) \tag{14}$$

where $\mathbf{h}$ is a vector-valued function receiving all polytope centers and returning the stack of all evaluations of h on each center. Since the support graph associated with W_γ is the complete graph and all entries are strictly positive, by the Perron-Frobenius theorem, the velocity vector with the gradient input, the formation velocity and the noise converges to consensus.

One can replace $c(k)$ by the true positions $q(k)$ by adding a second perturbation term whose norm is solely dependent on

the maximum estimation error ξ and the Lipschitz constant of h in the following manner:

$$v(k+1) = W_\gamma v(k) + \epsilon \nabla \mathbf{h}(q(k)) + \zeta v^f(q(k)) + \sigma^{\mathrm{p}}(k) + \sigma^{\xi}(k). \tag{15}$$

The above consensus dynamics will converge to a cluster velocity vector driving the cluster positions to points that have zero input signals of the gradient and $v^f(q(k))$. Let us first consider the case where the cluster is in a position where the term $\nabla \mathbf{h}(q(k))$ is negligible since the function $\mathbf{h}$ is not assumed to be strictly convex. Then, the consensus dynamics will converge to a neighborhood of size dependent on the norm of the noise signals $\sigma^{\mathrm{p}}(k) + \sigma^{\xi}(k)$ around a point such that $v^f(q(k))$ is zero, and the conclusion i) follows.

If on the other hand, the signal $v^f(q(k))$ is negligible in comparison with $\nabla \mathbf{h}(q(k))$, the consensus dynamics converges to a neighborhood around a local maximum as seen in [22], and ii) conclusion follows. ■

We remark to the reader that the results in Theorem 1 seem to conflict with some of the evidence in the simulations section. The value of $\delta = 0.08$ seems to point towards the case examined in Theorem 1. However, given that the range of h typically will be $[0, 2000]$ for the simulated cases, even a $\delta = 0.08$ is not sufficiently small to make the term $\delta \cdot u_i^{attr}(k)$ serve as a perturbation. This was done by design since the intended behavior is to have nodes move in loosely coupled formation when exploring the mission plane and then converge to a rendezvous point of largest utility, i.e., a single location that is a local maximum of h. The term $u_i^{attr}(k)$ when added in the velocity consensus dynamics can be seen as a gradient-free optimization and similar results are applicable to those ii) in Theorem 1.

VI. Simulation Results

To evaluate the proposed algorithm, multiple simulations across a variety of environments were executed. Due to space considerations, only the simulations for the fully-decentralized solution (Section IV) are presented. All simulation figures and videos are available on GitHub (`https://github.com/RafaelMenesesRibeiro/MasterDissertation`).

A. Simulation Goals

The simulations presented in this section are used to verify the following design objectives:

1) agents move towards and remain in the rendezvous areas while these are in the utility function;
2) agents are repelled by the undesired areas;
3) agents do not enter unauthorized areas;
4) agents do not leave the mission plane;
5) agents do not collide;
6) indefinite movement deadlocks do not occur;
7) agents eventually leave unimportant areas;
8) all agents move in every discrete time instant;
9) agents enter and remain in formation;
10) agents do not compete for formation slots.

B. Simulation Environment

Every simulation utilizes identical agents, including their control laws' weights, defined in (10): $\theta = 0.07, \beta = .01, \gamma = 0.6, \delta = 0.08, \epsilon = 1.0, \zeta = 1.0, \eta = 0.01$. Additionally, simulations generally use a default 100×100 mission plane size, and 10 agents with a sensing radius $SR = 15$ distance units. These parameters are modified in specific simulations when explicitly stated.

The simulations have a pre-imposed hard time limit (generally equivalent to 500 discrete time steps) to be able to be recorded and analyzed. The time limit can result in the configuration of agents presented in the final figure of a simulation to imply that some agents are in a deadlocked state or in positions with minimal utility value to the mission objective. This misinterpretation is primarily manifested in simulations with dynamic-valued rendezvous areas: the agents reach an area, explore it, the area is removed from the mission plane, and the final configuration illustrates the agents in an area with no rendezvous area. To mitigate this occurrence, each simulation is presented with multiple snapshots of configurations across the simulation duration. Finally, the simulation videos demonstrate that the behavior of the agents in the final time steps is as expected.

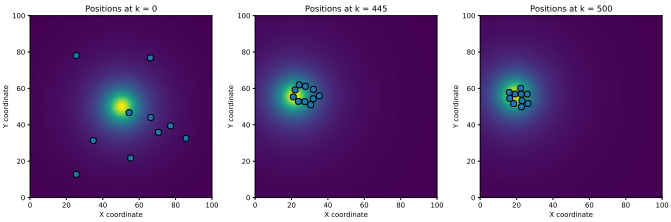
C. Analytical Analysis

The environments used are the following:

1) Single maximum utility function - to account for rendezvous missions:
   a) Static rendezvous area;
   b) Moving rendezvous area;
2) Single minimum utility function - to illustrate escape missions:
   a) Static minimum area;
   b) Moving minimum area;
3) Multiple maxima and minima - to depict the cases of conflicting objectives:
   a) Static rendezvous areas;
   b) Static rendezvous areas in a mission plane with unauthorized zones;
   c) Static rendezvous areas with dynamic values;
   d) Static rendezvous areas with dynamic values in a mission plane with unauthorized zones;
   e) Static rendezvous areas with dynamic values and random maxima/minima creation and destruction;
   f) Static rendezvous areas in a 200×200 mission plane with 30 agents.

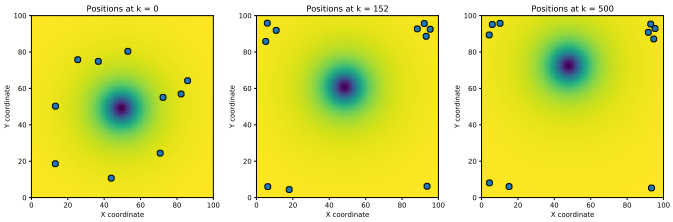
In the cases of multiple maxima and minima - case 3 - the minimum areas are also static, and in 3e, they are also created and destroyed randomly.

In order to represent the utility function, the simulation figures use a relative color scale: yellow represents positions with the highest utility value, and oppositely, purple corresponds to the lowest utility value. Positions with intermediate values have an intermediate color. As such, in simulations with rendezvous areas with dynamic values - directly correlated to a change in values of the utility function - the colors of



(a) Configuration at k = 0 (b) Configuration at k = 445 (c) Configuration at k = 500

Fig. 1: Agent configurations for a simulation with a single moving maximum, initially at the center of the mission plane. Using *Algorithm 2*.



(a) Configuration at k = 0 (b) Configuration at k = 152 (c) Configuration at k = 500

Fig. 2: Agent configurations for a simulation with a single moving minimum, initially at the center of the mission plane. Using *Algorithm 2*.

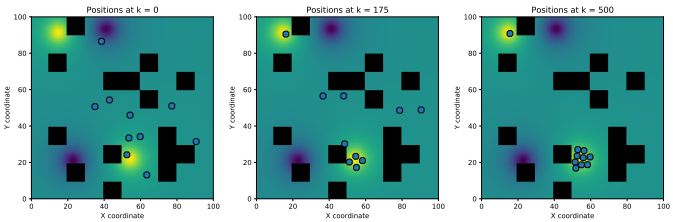
the mission plane's positions will change as the simulation progresses.

1) Single Rendezvous Area: Due to space constraints, figures from simulation type 1a are omitted. Figures 1a to 1c illustrate the scenario of a single randomly moving rendezvous area, initially at the center of the mission plane - type 1b. In the two types with a single maximum, the agents rendezvous on the desired area, and additionally, with a moving area, the agents dynamically track it and rendezvous in its new position. This first example demonstrates that convergence to a single cluster in the rendezvous area can be achieved from an initially disconnected network topology (objective 1). Figure 1b shows the agents creating a formation (objectives 9 and 10). The agents prioritize reaching the high-utility area rather than maintaining a formation, as it is demonstrated by the general absence of formations whenever the agents are at a rendezvous area (across all simulations). Objectives 4, 5, 6, and 8 are also realized, and will generally not be explicitly stated in the remainder of this section, because their fulfillment is constant across all simulations presented. Objectives 2 and 3 are invalid in this simulation type, and objectives 7 and 10 are not observable because the agents reach the objective too quickly.

2) Single Minimum Area: Due to space constraints, figures from simulation type 2a are omitted. Figures 2a to 2c show the simulation with a moving minimum area (type 2b). Due to the mission plane not having an explicitly defined rendezvous area, the agents do not form a single final cluster. These simulations demonstrate that the agents avoid the undesirable area, both when they are static or moving, (objective 2), and create formations (objectives 9 and 10): triangle, rhombus, and pair in Figure 2b.

3) Multiple Static Rendezvous Areas: The simplest scenario with multiple rendezvous and undesired areas - type 3a - results in convergence to the multiple static rendezvous areas while avoiding the undesirable (low-utility) areas (objectives 1 and 2) and maintaining formations (objectives 9 and 10). The simulation figures are omitted.

4) Multiple Static Rendezvous Areas in a Mission Plane with Unauthorized Areas: Figures 3a to 3c illustrate type 3b. Similarly to 3a, the agents converge to the rendezvous areas, while evading the low-utility areas (objectives 1 and 2). Additionally, the agents avoid the unauthorized areas represented



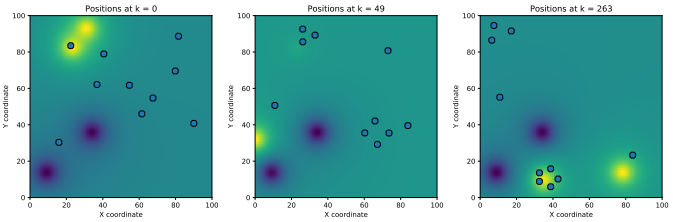
(a) Configuration at k = 0 (b) Configuration at k = 175 (c) Configuration at k = 500

Fig. 3: Agent configurations for a simulation with a simulation with a mission plane with multiple static maxima, minima, and unauthorized areas (black squares). Using *Algorithm 2*.

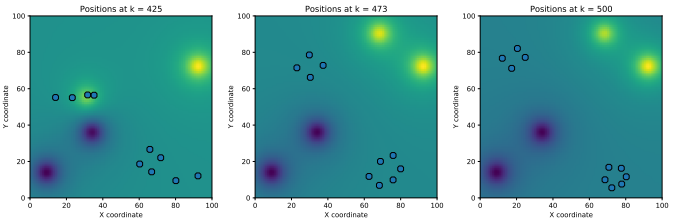
by the black squares (objective 3). Figure 3b shows the agents in formation: 4 agents at the bottom forming a square, and two pairs of agents in the middle, each forming line. It is unobservable in the still figures, but the paired agents maintain the relative position seen in Figure 3b to their counterpart, as shown in the simulation video.

5) Multiple Static Rendezvous Areas with Dynamic Values: Type 3c is illustrated in Figures 4a to 4f. The utility values of the rendezvous areas change based on the exploration of the agents. The agents rendezvous to the multiple desired areas, and when they detect the areas have been removed, they continue exploring the mission plane for new high-utility locations (objective 1). The cluster of five agents at the lower left in Figure 4c detects the rendezvous area has been removed (between 4c and 4d) and explores for a new one (4d, 4e, and 4f). Additionally, the agents avoid the low-utility areas (objective 2) and create formations (objectives 9 and 10), such as the rhombus and the triangle in Figure 4b, the pentagon and the same triangle in 4c, and the hexagon in 4f that started to form in 4d.

6) Multiple Static Rendezvous Areas with Dynamic Values in a Mission Plane with Unauthorized Areas: Type 3d creates an environment merging type 3b with 3c, with the results of the respective simulations persisting, totalling to objectives 1 to 6, and 8 to 10. Objective 7 is not considered to be achieved because the formation of six agents observed at the top of Figure 5c does not leave the area, as evidenced by 5d, 5e, and 5f. This is attributed to all the agents in the cluster

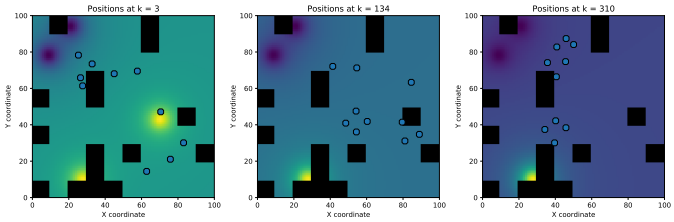


(a) Configuration at k = 0 (b) Configuration at k = 49 (c) Configuration at k = 263

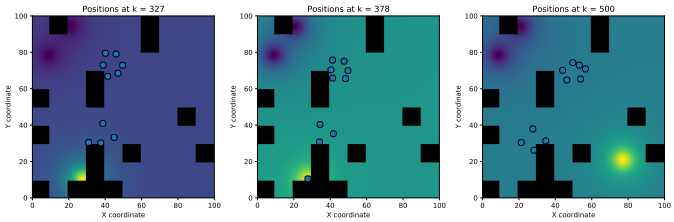


(d) Configuration at k = 425 (e) Configuration at k = 473 (f) Configuration at k = 500

Fig. 4: Agent configurations for a simulation with a mission plane with multiple static maxima with dynamic values and multiple static minima. Using *Algorithm 2*.



(a) Configuration at k = 0 (b) Configuration at k = 134 (c) Configuration at k = 310



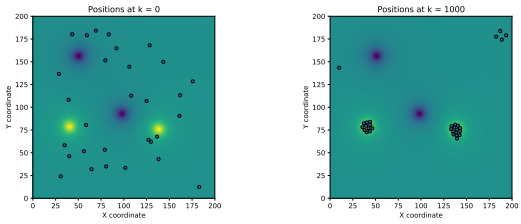
(d) Configuration at k = 327 (e) Configuration at k = 378 (f) Configuration at k = 500

Fig. 5: Agent configurations for a simulation with a mission plane with multiple static maxima with dynamic values, multiple static minima, and unauthorized areas (black squares). Using *Algorithm 2*.

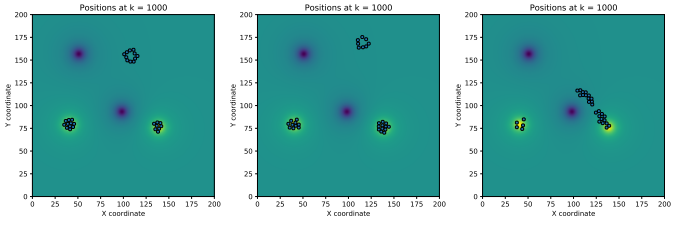
having the same utility values and being in a zero-gradient area. This results in their final control law being governed by the *Formation* component - which becomes the highest-weight component with a non-zero value (*Utility* and *Attraction* are estimated to be zero-vectors here). From the path of the lone agent at the lower rendezvous area in Figure 5e, from its position in 5c to 5e, this simulation shows the agents reach a desired area in the presence of static environmental obstacles in their path.

7) Multiple Static Rendezvous Areas with Dynamic Values and Random Creation/Destruction of Rendezvous Areas: In type 3e the agents' clusters converge to multiple rendezvous areas in the presence of a highly-dynamic utility function. All objectives (1 to 10) are achieved. The figures are omitted due to space concerns.

8) Multiple Static Rendezvous Areas in a 200×200 *Mission Plane with* 30 *agents:* Multiple simulations of type 3f are presented. The first simulation considers agents with a sensing radius, SR, of 15 distance units, depicted in Figures 6a and 6b. It is shown 25 of the 30 agents converge to the rendezvous areas, from the initially disconnected network topology. The limited rendezvous was initially attributed to the small sensing radius compared to the size of the mission plane. To verify this hypothesis, further simulations were executed from the initial configuration shown in Figure 6a, with increasing values for the agents' sensing radii. From Figures 6c to 6e, it can be seen that attributing the responsibility of imperfect rendezvous only to the sensing radius value would be inaccurate because with $SR = 45$, 300% of the initial value, full convergence is not achieved. With $SR = 60$, almost all agents reach a rendezvous area. With smaller SR values (15, 35, and 45), the partial rendezvous is attributed to the small sensing radius compared to the size of the mission plane (as already proposed) and to the agents' prioritization of creating and maintaining formations over leaving unimportant areas and exploring the mission plane.



(a) Configuration at k = 0 (b) Configuration at k = 1000 with $SR = 15$



(c) Configuration at k = 1000 with $SR = 35$ (d) Configuration at k = 1000 with $SR = 45$ (e) Configuration at k = 1000 with $SR = 60$

Fig. 6: Agent configurations for a simulation with a 200×200 mission plane with multiple static maxima and minima and 30 agents with the sensing radius indicated in the captions (15, 35, 45, 60). Using *Algorithm 2*.

D. Results Discussion

Objectives 4, 5, 6, and 8 were verified across all simulation scenarios presented, and as such, are considered fully accomplished.

The simulation types 2a and 2b serve to test if the agents are repelled from undesired areas, either statically-positioned or randomly moving. From the simulation figures, it can be observed that this objective - objective 2 - is fully accomplished.

Analogously, the primary objective of types 3b and 3d is to verify the effectiveness of the agents' environmental obstacle collision avoidance method. Due to the implemented method not providing exact collision detection with obstacles, agents partially intersect unauthorized areas, objective 3 is considered partially accomplished. This can be observed in Figure 5b, where an agent of the triangular formation on the lower right is intersecting the square-shaped unauthorized area. It is left for future work the implementation of an environmental obstacle collision detection that provides exact collision results based on the method used to detect collisions between agents: ray casting between an agent's polytope and the polytopes of the obstacles in the agent's vicinity.

Agent formations can be observed across all simulations presented. Moreover, the formation structures change as the simulation progresses depending on the number of agents participating: Figure 4d illustrates two formations on the right side, a pair and a rhombus; these two formations are then merged (evidenced by Figure 4e) to form a hexagon in Figure 4f. There is no observable evidence the agents compete for formation slots. Therefore, objectives 9 and 10 are considered fully accomplished. To improve the efficiency of the agent formations, it is left for future work the development of a heuristic that minimizes the global cost for the agents to move to their assigned formations slots. The current implementation is based on a greedy approach, which can result in unbalanced movement distances (to the formation slots) between the participating agents.

The simulation presented in Section VI-C.6 illustrates agents that remain in the same low-utility area for approximately 40% of the simulation. This is attributed to the *Formation* component of the agents' movement, which is the component with the highest weight, matched by *Utility*: as the agents are in a zero-gradient area (evidenced by the uniform color in their positions), they prioritize creating a formation. The presence of statically-positioned environmental obstacles in proximity with the cluster can contribute to the absence of exploration. However, this phenomenon is additionally observed in simulation type 3e (Section VI-C.7), which does not contain unauthorized areas. Consequently, objective 7 is considered partially accomplished - the agents generally eventually leave unimportant areas when they are not prioritizing maintaining formations. It is left for future work the development of dynamic weights for the movement components in the desired movement law. The weights would change based on the agent's current situation: if an agent has been in the same low-utility area for more than a finite limit, the weight of the *Randomness* component is increased; or if an agent is at a rendezvous area, the *Utility* component's weight is decreased - to allow for the creation of formations in rendezvous areas.

The agents' primary objective of achieving rendezvous to desirable areas was a focus on all simulations, with the agents always reaching rendezvous areas (also considering the implicitly-defined highest-utility areas in the simulations with a single minimum). In the simulations with a single rendezvous area (Section VI-C.1), all the agents achieved consensus on the rendezvous area. In the simulations with a single undesired area (VI-C.2), the agents avoided the undesired area, remaining in positions with the highest utility values in the scenario. In simulations with multiple rendezvous and undesired areas (VI-C.3 to VI-C.8), the agents reached the rendezvous areas in the presence of environmental obstacles, random area destruction, and high-utility area removal due to agent exploration. Consequently, objective 1 is considered fully accomplished.

VII. Conclusions and Future Work

This paper focused on the rendezvous problem for a group of mobile agents attempting to converge to multiple dynamic targets in a decentralized system while creating and maintaining formations. The agents are assumed to be equipped with localization and measuring sensors to be aware of their position, velocity, and utility value, and those of their neighbors. The proposed solution does not require communication nor the network topology to be connected. Instead of a consensus-based approach, the behavior of the agents uses flocking-based movement rules, with added components tailored to our scenario. We also introduced a mechanism, with theoretical guarantees, to avoid collisions between agents and a mechanism to avoid collisions with environmental obstacles. The system considers imperfect measurements by working with set-valued estimates of the real positions, defined to be convex polytopes. It was shown through multiple simulations, which represent a multitude of scenarios with real-world applications, the effectiveness of our algorithm, and its limitations. These empirical results showed the agents rendezvous to the multiple dynamic rendezvous areas in the presence of undesirable areas, static environmental obstacles, and arbitrary changes in the utility function, with varying degrees of efficiency.

For a possible future work direction, we consider the propagation of the SVOs, used for updating the set-valued estimates, and apply Model Predictive Control strategies to better decide on the actuation.

References

[1] R. Ribeiro, D. Silvestre, and C. Silvestre. Decentralized formation control for multi-agent systems based on flocking rules. *IEEE Transaction on Control of Network Systems*, under review.

[2] R. Ribeiro, D. Silvestre, and C. Silvestre. Decentralized control for multi-agent missions based on flocking rules. In José Alexandre Gonçalves, Manuel Braz-César, and João Paulo Coelho, editors, *CONTROLO 2020*, pages 445–454, Cham, 2021. Springer International Publishing.

[3] R. Ribeiro, D. Silvestre, and C. Silvestre. Partially decentralized formation control based on flocking rules. *Systems & Control Letters*, under review.

[4] R. Olfati-Saber and R.M. Murray. Consensus problems in networks of agents with switching topology and time-delays. *IEEE Transactions on Automatic Control*, 49(9):1520 – 1533, September 2004.

[5] D. Antunes, D. Silvestre, and C. Silvestre. Average consensus and gossip algorithms in networks with stochastic asymmetric communications. In *50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC)*, pages 2088–2093, Dec 2011.

[6] D. Silvestre, J. P. Hespanha, and C. Silvestre. Broadcast and gossip stochastic average consensus algorithms in directed topologies. *IEEE Transactions on Control of Network Systems*, 6(2):474–486, June 2019.

[7] S. Patterson, B. Bamieh, and A. El Abbadi. Convergence rates of distributed average consensus with stochastic link failures. *IEEE Transactions on Automatic Control*, 55(4):880 –892, April 2010.

[8] F. Fagnani and S. Zampieri. Average consensus with packet drop communication. *SIAM Journal on Control and Optimization*, 48(1):102–133, 2009.

[9] R. Carli, F. Bullo, and S. Zampieri. Quantized average consensus via dynamic coding/decoding schemes. *International Journal of Robust and Nonlinear Control*, 20(2):156–175, 2010.

[10] D.V. Dimarogonas and K.H. Johansson. Event-triggered control for multi-agent systems. In *48th IEEE Conference on Decision and Control, held jointly with the 28th Chinese Control Conference (CDC/CCC)*, pages 7131 –7136, December 2009.

[11] D. V. Dimarogonas, E. Frazzoli, and K. H. Johansson. Distributed self-triggered control for multi-agent systems. In *49th IEEE Conference on Decision and Control (CDC)*, pages 6716 –6721, December 2010.

[12] R. Parasuraman, J. Kim, S. Luo, and B. Min. Multipoint rendezvous in multirobot systems. *IEEE Transactions on Cybernetics*, 50(1):310–323, Jan 2020.

[13] A. Turgeman, A. Datar, and H. Werner. Gradient free source-seeking using flocking behavior. In *American Control Conference (ACC)*, pages 4647–4652, July 2019.

[14] A. Singha, A. Ray, and A. Samaddar. Trajectory tracking in the desired formation around a target by multiple uav systems. *Procedia Computer Science*, 133:924–931, 01 2018.

[15] H. Qiu and H. Duan. Multiple uav distributed close formation control based on in-flight leadership hierarchies of pigeon flocks. *Aerospace Science and Technology*, 70, 08 2017.

[16] X. Fu, J. Pan, H. Wang, and X. Gao. A formation maintenance and reconstruction method of uav swarm based on distributed control. *Aerospace Science and Technology*, page 105981, 06 2020.

[17] R. Olfati-Saber and R. Murray. Distributed cooperative control of multiple vehicle formations using structural potential functions. *IFAC Proceedings Volumes*, 35, 07 2002.

[18] F. Mehdifar, C. Bechlioulis, F. Hashemzadeh, and M. Baradarannia. Prescribed performance distance-based formation control of multi-agent systems. *Automatica*, 119:109086, 09 2020.

[19] D. Silvestre, P. Rosa, J. P. Hespanha, and C. Silvestre. Set-consensus using set-valued observers. In *American Control Conference (ACC), Chicago, Illinois, USA.*, July 2015.

[20] R. Ribeiro, D. Silvestre, and C. Silvestre. A rendezvous algorithm for multi-agent systems in disconnected network topologies. In *2020 28th Mediterranean Conference on Control and Automation (MED)*, pages 592–597, Sep 2020.

[21] D. Silvestre, P. Rosa, J. P. Hespanha, and C. Silvestre. Stochastic and deterministic fault detection for randomized gossip algorithms. *Automatica*, 78:46 – 60, 2017.

[22] H. Mohammadi, M. Razaviyayn, and M. R. Jovanovic. Robustness of accelerated first-order algorithms for strongly convex optimization problems. *IEEE Transactions on Automatic Control*, pages 1–1, 2020.